A Blockchain-based Decentralized Machine Learning Framework for Collaborative Intrusion Detection within UAVs

Ammar Ahmed Khan[a], Muhammad Mubashir Khan[a], Kashif Mehboob Khan[b], Junaid Arshad[c], Farhan Ahmad[d]

[a]*Department of Computer Science & IT, NED University of Engineering & Technology Karachi, Pakistan*
[b]*Department of Software Engineering, NED University of Engineering & Technology Karachi, Pakistan*
[c]*School of Computing and Digital Technology, Birmingham City University, Birmingham, UK*
[d]*Systems Security Group, Institute for Future Transport and Cities, Coventry University, Coventry, UK*

Abstract

UAVs have numerous emerging applications in various domains of life. However, it is extremely challenging to gain the required level of public acceptance of UAVs without proving safety and security for human life. Conventional UAVs mostly depend upon the centralised server to perform data processing with complex machine learning algorithms. In fact, all the conventional cyber attacks are applicable on the transmission and storage of data in UAVs. While their impact is extremely serious because UAVs are highly dependent on smart systems that extensively utilise machine learning techniques in order to take decisions in human absence. In this regard, we propose to enhance the performance of UAVs with a decentralised machine learning framework based on blockchain. The proposed framework has the potential to significantly enhance the integrity and storage of data for intelligent decision making among multiple UAVs. We present the use of blockchain to achieve decentralized predictive analytics and present a framework that can successfully apply and share machine learning models in a decentralised manner. We evaluate our system using collaborative intrusion detection as a case-study in order to highlight the feasibility and effectiveness of using blockchain based decentralised machine learning approach in UAVs and other similar applications.

Keywords: Unmanned Aerial Vehicles, UAV, Blockchain, Machine Learning, Decentralized Machine Learning, Collaborative Intrusion Detection

1. Introduction

Unmanned Aerial Vehicle (UAV) or drones is a type of aircraft which is operated by an autonomous computer-based pilot instead of a human on board. Initially, UAVs were primarily used by military for training soldiers or attacking the enemies while safeguarding the life of human pilots [1]. Over the time there have been immense innovations in the development and capabilities of UAVs that has created numerous interesting applications of drones other than military. Latest UAVs are either remotely controlled by human or by an intelligent computer based autopilot system which has induced many of their interesting applications in a variety of domains. These applications include disaster relief [2], road safety [3], transport and delivery of medicines [4], agriculture [5], recreation [6], archaeology [7] etc. These applications have attracted a lot of new requirements with many innovative ideas of utilising UAVs. Amazon Prime Air is the latest example delivery drones in selected areas of United States [8]. Similarly, the Federal Aviation Administration (FAA) of United States has authorised several drone delivery companies to operate in remote areas for delivering life saving medicines and blood especially during the COVID-19 pandemic [9].

Majority of the UAVs applications can threaten public safety, airspace security and national defence systems. Consequently, UAVs require better control, management, communication, data storage, and intelligent decision making without any delays [10]. These requirements become more critical when a coordinated operation of large number of UAVs is desirable with collaborative and intelligent decision making in a decentralised and distributed manner [11]. UAV systems mostly depend upon the centralised server or a cloud based platform to perform processing of data with complex Machine Learning (ML) algorithms. In fact, all the conventional cyber attacks are also applicable for the UAVs while their impact would be more serious because UAVs are highly dependent on smart systems that extensively utilise artificial intelligence and machine learning techniques in order to take decisions in human absence. Machine learning holds great promise as a solution to a variety of challenges caused by vast amount of data generated by UAVs and other cutting-edge technological systems [12], real-time astronomical data [13], Internet of Things (IoT) [14] and Cyber-Physical Systems (CPS)[15]. ML techniques enable intelligent processing of large volumes of complex datasets to identify patterns with minimal human intervention which

is extremely important for the operation of UAVs and other autonomous systems where complex decision making and coordination is required. Such techniques are indeed well-established and have been used extensively to address variety of other problems including customer churn analysis [16], self-driving vehicles [17], intrusion detection [18], speech recognition [19], data loss prevention [20], healthcare [21] and consumer behaviour prediction [22]. Due to extremely complex nature of many such problems where an algorithmic solution becomes infeasible, it is important to explore machine learning-based decision making solutions achieved through intelligent analysis of large datasets.

However, the effectiveness of machine learning techniques to achieve high accuracy of predictive analytics typically requires large volume of training dataset [23]. In addition, with the emerging advancements in UAVs with new application domains requiring huge datasets are often spatially-distributed while being supplied by numerous parties that can be mutually untrustworthy [24]. For instance, UAVs having IoT devices continuously sense the environment and may contribute their data so that a unified global model can be constructed. However, such UAVs with IoT devices can typically be large in number with concerns such as privacy associated with sharing of data in its original form. In such scenarios, sharing of knowledge (through machine learning models) has been promoted to achieve high accuracy without compromising privacy of data.

Contemporary ML is usually centralized i.e. data from different devices scattered across the network is uploaded to a central server for model training and development. Once the model is trained, it is shared with all the devices to be used independently. This approach has its problems with data scalability owing to rising process impediment created by the learning process. A centralized solution also raises security and privacy challenges [25] such as trustworthiness between the contributing parties and central endpoint so that the resulting model from the learning process can also be trusted. There are applications such as real-time weather or astronomical data [13] and other mission critical applications in which huge data volumes are scattered across a large distributed system. In such scenarios, it is not cost-effective and reliable to gather data in a central location. Therefore, the choice of machine learning algorithm becomes extremely important to ensure data distribution, resilience to failures and parallel computation. Ideally, such a decentralized context requires a decentralised and distributed system to facilitate the learning process [26].

Current research and practice within machine learning is progressing towards performing ML by transferring model parameters rather than data to a central coordinator to construct a refined/reinforced model. In this respect, one approach is to use centralized blockchain-based collaborative mechanism for defending against cyber-attacks. Although such approach uses collaboration among participating nodes, it does not benefit from blockchain's inherent properties i.e. decentralization and is therefore susceptible to single point of failure. For instance, a centralized system in this scenario would have a different approach to consensus whereby machine learning models learned by the individual nodes could be ranked by an arbitrary central authority. An alternate methodology is the federated learning system in which more than one parties jointly learn the ML system such as, the deep neural network, with localised private and confidential data. Shokri and Shmatikov [27] and McMahan et al. [28] presented the proof of concept of such a system by applying differential privacy. However, it was later shown by Hitaj et al. [29] that employing such a record-level differential privacy becomes unsuccessful in a federated learning systems as addition of noise may distort results in differential privacy scheme.

To address the aforementioned challenges, we explore the use of blockchain technology to facilitate a method of achieving decentralized, distributed analytics whilst protecting the privacy of data and ensuring the trustworthiness of the process. Blockchain has received significant attention in recent years primarily due to its benefits such as immutability of data, decentralisation, distributed design, transparency, and trustless consensus [30]. Consequently, blockchain has been adopted by range of applications including UAVs [31][32], supply chain [33], digital asset management [34], e-government[35], intrusion detection [36] and many more. The immutable nature of blockchain lends itself to diverse application domains requiring robustness and better audit trails [37].

Our research is inspired by the concept of federated machine learning [38] where data sharing takes place by ensuring user privacy as part of the process. Privacy in data sharing can be factored in a technique that can ensure anonymity in the data by changing it all together without affecting the overall learning process [39]. We developed a blockchain-based decentralized machine learning scheme for collaborative intrusion detection, where we create a blockchain network of multiple nodes that perform independent intelligent analytics using custom choice of machine learning algorithm. We assume a distributed scenario where data is generated and collected in the form of distributed data streams such as UAVs. In such scenarios, collating data at a centralized server can incur performance overhead as well as posing risks to data privacy. Therefore, instead of sharing data across participating nodes, machine learning models are shared across them so as to use this knowledge to improve accuracy of intelligent analytics across participating nodes.

In order to assess the effectiveness of our proposed scheme, we use the challenge of intrusion detection in a system like UAVs a use-case. In particular, we used KDD99 attack dataset, splitting 10,000 rows across

participating nodes of the blockchain network providing coverage of different attack types. We measure accuracy, false positive rate, true positive rate, recall, precision, and F1 score at each node both before and after sharing the models.

The major contributions of this paper are summarised as follows:

1. Design and development of a generic blockchain system to aid decentralized, peer-to-peer machine learning-based analytics. The proof of concept presented in the paper enables sharing of machine learning models without the need to share data used to develop this model thereby achieving collaboration among nodes without compromising privacy of the participating nodes.

2. Evaluation of the proposed system within a collaborative intrusion detection scenario. Experimentation with the proof of concept has been conducted with the KDD99 attack dataset to evaluate its ability to support collaboration among distributed nodes to achieve effective intrusion detection. In this regard, collaborative knowledge sharing is achieved across participating nodes by using stacking to reinforce machine learning models developed at individual nodes.

Rest of this paper is organised as follows. Section 2 presents the fundamental background knowledge about blockchain technology highlight its primary concepts followed by Section 3 which presents an account of the existing work in this domain. Section 4 presents the design and implementation of our proposed system to achieve decentralized machine learning. Evaluation of the proposed system within intrusion detection use-case is presented in Section 5 which highlights the performance of the proposed system with respect to its effectiveness for intrusion detection. Section 6 concludes this paper.

2. Blockchain Technology

Blockchain technology has attracted significant attention primarily due to its success as the technology underpinning Bitcoin. The fundamental concept at the core of blockchain is that of a distributed, decentralized ledger whereby a blockchain network is run by the peers or participating nodes without any centralized authority [40]. Further, the participation of nodes (number and type) is driven by the type of application. After the remarkable success of blockchain as the driving technology of famous cryptocurrencies including the Bitcoin and Ethereum a large number of other applications of blockchain have been realised in the past few years. These non-cryptocurrency applications of blockchain include several interesting UAV applications such as blockchain-based smart vehicular networks [37], secure UAV data sharing [30], privacy preservation in 5G
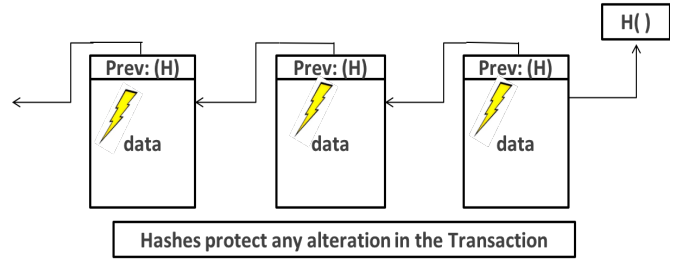

Figure 1: Tamper resistant blockchain

enabled UAVs [41], secure routing of swarm UAV networking [42] and UAV path planning for healthcare [43].

With respect to the ledger, a **transaction** represents the most important concept within blockchain. A transaction in blockchain is a piece of information which moves something of value (a digital token which may represent a unit of currency, a vote etc.) from one public address (belonging to the sender) to the receiver's address. Therefore, a transaction saves and tracks the state of the blockchain over the period of time. These transactions become the part of blockchain forever through blocks which move them into chain. A **block** is primarily a collection of transactions which are integrated and organized in such a way that each block computes and keeps its own blockhash (using the individual hashes of all the transactions as its source) in the block along with the blockhash of its preceding block. In this way, a chain of blocks is generated which increases with time. Since these blocks are connected through their hashes (computed through the transactions within that particular block), this data structure makes the records of blockchain immutable where a slight change in a single transaction would produce an entirely new hash resulting in a mismatch of this hash with the neighbouring block. This prevents any suspicious block to be accepted by the blockchain network and therefore mitigates against illegitimate tampering of blockchain state. Figure 1 demonstrates this linkage between different blocks to achieve a tamper-resistant ledger.

All the nodes of a typical blockchain network store an identical copy of blockchain locally which is frequently synchronized with the main blockchain (also known as *consensus blockchain*). Each new block of a blockchain is accepted and added by its peers through a process known as **mining**. The process of mining is essential in developing consensus among participating nodes and can take up different forms depending upon the type of application.

2.1. Consensus in Blockchain

Due to the decentralized nature of the blockchain network, the mechanism to achieve consensus is critical in achieving a trustworthy, tamper-proof ledger. Due to the variety of applications using blockchain, there are

various approaches in use for maintaining consensus in blockchain which can be broadly divided into two categories i.e. consensus achieved through an evidence, and voting based consensus (Byzantine Fault Tolerance and Crash Fault Tolerance based consensus). The evidence based consensus is one of the most popular approaches due to the use of *Proof of Work (PoW)* by Bitcoin however other variants such as Proof of Stake have also emerged.

Proof of work is the most common consensus based algorithm in the blockchain applications [44]. This scheme relies heavily on the computational capabilities of the mining hardware to solve a non-trivial mathematical problem. Although all participating nodes can attempt to solve this mathematical challenge, the hashing power of the miner acts as a decisive factor to solve the challenge in a timely manner. The mathematical problem to be solved usually involves searching for a number (known as nonce value) which when hashed (after appended to the given data) should produce a value less than a specific value that is the current target of the blockchain network. In recent years, PoW in particular and proof based consensus algorithms in general have been extended to address their constraints with respect to security (such as the likeliness of carrying out double spend attack [45]) and performance efficiency [46].

2.2. Cryptography in Blockchain

Blockchain makes use of asymmetric cryptography to ensure the privacy in the exchange of messages between a sender and a receiver [47]. This public-key cryptography technique also allows every participant to verify the transaction. Since the public key is available to everyone, the sender uses this key to send a message (for asset transference) to the intended receiver which can only be decrypted through the receiver's private key. Consider 3 blocks as shown in Figure 2 such that Block 1 has the information of P1 with the hash value of Q1, Block 2 has the information of P2 with the hash value of Q2 while Block 3 has the information of P3 with the hash value of Q3. Q3 is created from the combination of Q2 and P3 and so on while P1 comes from the default value P0. Now if someone changes the hash values like P2 to P21 and Q2 to Q21 and other blocks are kept same as earlier, in this case it represents an unstable blockchain. Therefore to make the blockchain stable, the values of Q3 will have to be changed to Q31 along with P2 to P21.

2.3. Public vs. private blockchain

From the perspective of participation of nodes, a blockchain can be divided into two broad categories; public and private blockchains. **Public blockchain** [48], as the name suggests, adopt a public model for participation and therefore may be joined by any node without any restriction. Such networks of blockchain do not require any permission for a user to join or participate in the network. As mentioned earlier, it is a permissionless open-ended blockchain and that is why the network size is usually bigger than the permissioned blockchain. **Private blockchain** also called *permissioned blockchain*, on the other hand, is a more controlled form of blockchain which is not publicly accessible. In a private blockchain [49], nodes must seek permission to join the network. Such networks usually require an authenticated node to perform according to a predefined role in the system.

3. Related Works

The widespread use of machine learning techniques and algorithms in diversified applications has highlighted many challenges in terms of limited processing and storage capacity against large training datasets. For example, in the case of artificial neural networks, the storage and processing requirement may significantly increase when there are larger sets of input parameters to train the model. UAVs are an ideal use-case of such problem domains where individual devices are constrained with respect to resources available and therefore unable to host traditional, resource-hungry machine learning mechanisms.

With respect to federated learning, Bonawitz et al. [50] describe the federation concept in machine learning through a practical use case. They generate machine learning models on mobile devices where the data actually resides and the individual data from the model is combined in the cloud for the global model where a deep neural network is trained by using TensorFlow [51]. the authors have identified and addressed specific implementation concerns such as localization (time-zone), training of machine learning models based on device availability and constrained compute resources of the devices. The devices are in communication with the server that is responsible for storing the updated model. The latest model is pushed down to the device that updates it after performing model training and sends it back to the server. The communication frequency, device participation and device selection for a Federated learning task is based on a protocol that is planned for robustness in non-reliable conditions. It uses an analytics process to gain insights into bottlenecks and other other constraints occuring at the device end and proves to be helpful as the server doesnt need to have access to the device's data.

Konečný et al. [52] performed machine learning optimization using various algorithms. Interestingly, the optimization was carried out on different mobile devices having a portion of the data representing the users' own patterns. Their work primarily focuses on optimization and uses federation learning to improve the machine learning model. Specifically, authors focus on the challenges in achieving optimal performance with existing algorithms within a distributed learning environment.
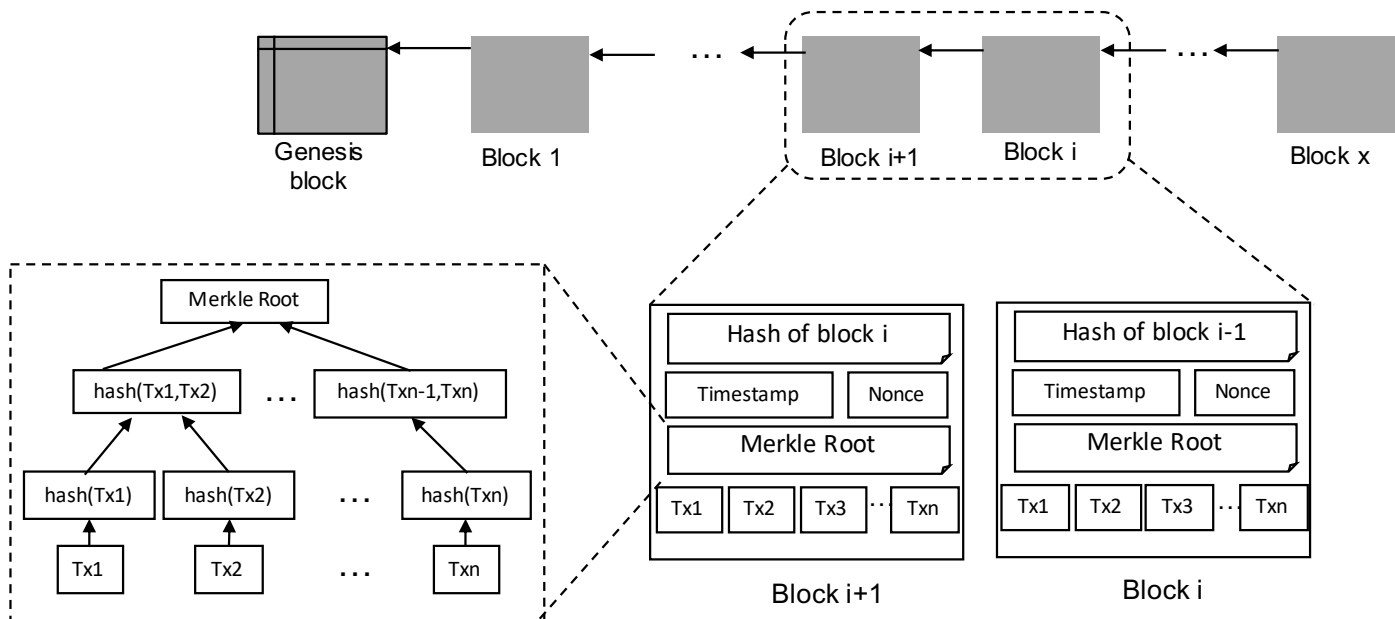
Figure 2: An in-depth view of a conventional blockchain

The two distributed machine learning algorithms that are discussed here include the Stochastic Variance Reduced Gradient (SVRG) [53] and the Distributed Approximate Newton (DANE) [54] which are used for the purpose of achieving optimization in a distributed environment. The authors have demonstrated that in this setting, traditional machine leanring models dont perform well due to their inherently sequential nature. The traditional algorithms incorporate fast iteration rounds whereas here, a communication round has been inserted in the learning process as the nodes exchange the learned parameters with the server responsible for updating the model. They further show with data achieved through experimentation that given the constraints of network bandwidth and challenges like slow convergence , it is possible to design an algorithm that works well in this setting.

M. Li et al. [55] presented one of the major contributions towards the distributed machine learning. The main issues addressed in their contribution are the technique of distributing training data and the workload over numerous worker nodes. Furthermore, the proposed framework is able to enable the server nodes for maintaining the globally shared model parameters in terms of sparse vectors. The data scale in consideration here is of the order of trillions of samples and possibly very high feature lengths. Servers that maintain a copy of the latest updated parameters are termed as parameter servers and worker nodes have a portion of the whole data on which they train. The machine learning models used by the authors are Sparse Logistic Regression and Latent Dirichlet Allocation. The system performed well due to decrease in communication cost due to bulk communication server and message compression on the transmission of parameter key,value.

T. Kraska et al. [56] presented an effort focused on collecting dataset from distributed nodes with each node performing its individual training algorithm. The focus in this system is to facilitate data pre-processing tasks including feature selection, engineering and optimization based on factors such as which model to use, what features to select and what time is expected to be taken by the given configuration. The proposed framework in this paper is also inspired with this model. However, in our proposed framework, we provide this functionality for the purpose of emulating the human behaviour of pre-processing the data, model training and creating a deployment-ready machine learning model. Furthermore, we use blockchains to enable trusted sharing of models and their performance indicators to enable privacy-aware federated learning in a trust-less environment.

Recently, there have been several interesting contributions which exploit the power of blockchain to achieve maximum benefit from machine learning algorithms. Kurtulmus et al. [57] demonstrate the use of *smart contracts* to evaluate and validate machine learning models which is performed by the smart contract. Once a user submits a dataset to the blockchain, they can then set a reward amount on that in ethereum tokens since the blockchain employed is ethereum-based. The various blockchain nodes participate in performing machine learning and once the machine learning task is completely by a human driven agent, the machine learning model is submitted to the smart contract that validates the solution. Once all the participants have successfully submitted their solutions, the best model wins the amount. In our scenario, we have emulated this behavior on the nodes through

automating the machine learning steps of pre-proicessing and model-fitting. The authors have created a prototype or a basis for a future market place where companies can get the best available machine learning expertise by submitting the dataset on the blockchain and rewarding the best model submission based on results. The authors have also implemented trust and fairness controls through smart-contracts in addition to model validation and reward. Furthermore, the authors have identified the risks related to cheating by the model submitters and the organizer and have controlled them by enforcing policies through smart contracts. Also, the machine learning process in this work has also been implemented in the solidity programming language. The language has many constraints and hence the authors performed several performance and memory-saving techniques. C. Xu et al. [58] present the intelligent datacenter's energy and resource management through blockchain. Specifically, smart contract performs machine learning to achieve the goal of further minimizing the energy cost of the datacenter. The output of the machine learning model is then used by the smart contract that performs tasks to control datacenter resources.

With respect to use of blockchains and machine learning within cyber security, a number of efforts have been made. For instance, Outchakoucht et al. [59], focus on the challenge of access control within Internet of Things (IoT) and leverage blockchains and machine learning to achieve a trustworthy, self-adjusting solution which can be continuously updated through the use of reinforcement learning. Dey et al. [60] proposed a methodology where machine learning is employed in a blockchain network to detect anomalies such as collusion that may lead to majority attack. Supervised machine learning and algorithmic game theory is used to take early action to counter the attack. X. Chen et al. [61] utilize Ethereum smart contracts to share locally learned gradients. In order to achieve privacy-aware data processing, authors use differential privacy whilst segregating data storage from data processing nodes.

With the increase in use of autonomous vehicles and UAVs, use of machine learning and blockchains to aid novel solutions within this domain is attracting significant attention. However, significant efforts within this domain relate to the use of machine learning to aid efficient and effective detection of UAVs. In this respect, [62], [63], [64] and [65] represent recent efforts to employ machine learning for effective UAV detection and classification. With respect to the use of blockchains within UAVs, [66], [67], [68] and [69] represent recent efforts which utilize blockchains to facilitate trustworthy applications within UAVs.

This paper is focused at exploring the use of blockchains to facilitate trustworthy distributed machine learning solutions within UAVs. In particular, the proposed framework adopts a federated machine learning strategy to leverage the distributed architecture of a typical UAV-based system whilst reducing the performance overheads incurred due to traditional centralised machine learning approaches. Further, the proposed framework uses blockchain technology to enable trustworthy sharing of machine learning models and associated metrics to enable an ensemble learning approach. We assess the effectiveness of the proposed system by implementing an intrusion detection scenario to identify potential advantages of the proposed approach as well as open challenges which require further work.

4. A Blockchain-based Decentralized Machine Learning Framework for UAVs

In this section, we present the architectural details of the proposed system followed by its implementation with blockchain technology.

4.1. System architecture:

The proposed system is designed to simulate a scenario where multiple distributed nodes are engaged in intelligent processing of data independently before collaborating to enhance accuracy of predictive analytics. For instance, intrusion detection within a CPS/IoT system is inherently distributed as the task of detecting misuse patterns is devolved to individual sensor nodes followed by collation of knowledge shared by individual nodes to achieve effective detection. A generic blockchain system was designed and implemented designed and implemented to provide a testbed for rapid deployment of scenarios involving blockchain and machine learning. As explained in Section 4.1 (system architecture), the blockchain and machine learning processes were loosely coupled and the machine learning component was embedded in the blockchain daemon by use of APIs whose design was inspired by the Python scikit machine learning library. Due to this API flexibility, stacking and non-stacking scenarios could be deployed quickly. The use of smart contracts to execute functionality was very important in our design. Smart contracts were written in such a way so as to aid the diverse use of machine learning API as well as data distribution. They also serve an additional purpose of data logistics, such as metadata input handling and node to node data transfer.

A high-level architectural diagram of the system is presented in Figure 3. Overall, the system architecture consists of a monitor node, multiple miner nodes, global address space, and the machine learning engine (further details of these are presented below). Both the monitor and miner nodes take part in the mining and machine learning process and are connected to each other to achieve inter-node communication.

4.2. Monitor node:

As with any other collaborative environment, the monitor node acts as the coordinator which is tasked
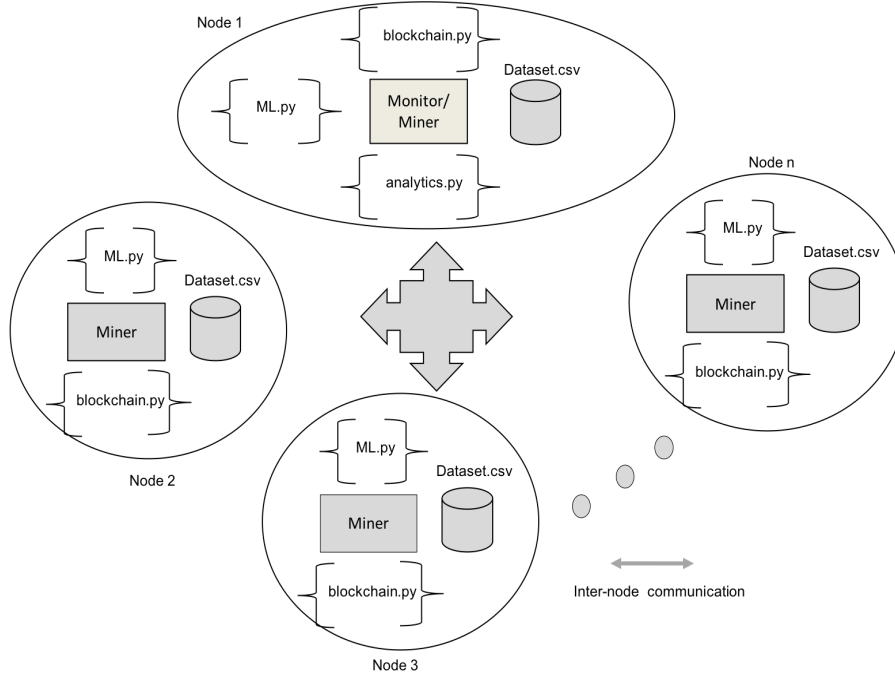
Figure 3: A blockchain-based decentralized machine learning system

with communicating with other nodes to facilitate achieving the desired outcome. It has its own global memory space and exposes this data through micro-services calls.These micro-service calls refer to the different functions of the blockchain daemon broken up into RESTful style lightweight services. the input to these services requires very little overhead and their main purpose is for internal communication, protocol and polling between the nodes. e.g. the command to initiate the machine learning process on all the nodes is one such micro-service call.. The IP of this system is static and is used by all other nodes to connect to the network much like Bitcoin miners that connect to the Bitcoin network through *getinfo* [70]call service. Within the context of a blockchain setting, monitor node performs the role of a *seed node*. Being the coordinator, there are number of tasks which a *seed node* performs in a blockchain network. These include admitting new miners to the network and Domain Name Service (DNS) for the mining nodes. Within our system, the monitor node performs these functions to adopt the working of Bitcoin network [71]. The monitor node is responsible for these functions within the proposed system

In order to aid experimentation with decentralized machine learning, an important phase is distributing data to be processed across all participating nodes. In a real-life scenario, we expect participating nodes to gather data using independent streams however in our proof-of-concept, the monitor node performs this function. Therefore, the monitor nodes distributes chunks of data to each mining node along with accompanying metadata which is envisaged to help the

machine learning engine. However, the monitor node is not only the coordinator and also participates in the machine learning and blockchain mining processes similar to other (miner) nodes of the network. This is highlighted in Figure 3 by assigning both monitor and miner node roles to the coordinating/seed node.

4.3. Miner nodes:

Miner nodes use machine learning algorithms to generate independent analytical models by intelligent processing of dataset available to them. As highlighted in Figure 3, the analytics.py performs operations required for specific ML algorithms. These miners in a traditional blockchain network choose transactions, mine them into a block and add them to the blockchain using proof of work or other consensus algorithms. In our scenario, the miners are utilizing their computing resources to achieve accuracy in their machine learning model against a dataset.

4.4. Global address space:

Similar to a Distributed Hash Table (DHT), our prototype features a list of nodes that are actively connected to the network. This global addresses memory space stores the network and node addresses in the following format.

$$<IP> : <port>$$

The nodes list is persisted in a global memory space that is easily accessible by other nodes simply through micro services calls. Miners can carry out their mutual

7

coordination through this global address space. They can also choose to have a copy of this global address space that is updated every time a new node is added to the network. In our implementation, we have decided to allocate each node a copy of this address space that gets updated on each node as the network grows. The same is done in Multichain [72], except here, each node pings the monitor node and gets the others added. In Multichain, the first node returns an IP and port and this network address is provided to new nodes as they subscribe to the blockchain network.

4.5. Machine learning engine:

In order to achieve decentralized machine learning setup, each miner node has an embedded ML engine which interfaces with the blockchain component of the node. These components exchange information throughout the life-cycle of machine learning analytics process using the available dataset. The current design envisages that each node has the liberty to choose its own machine learning model. Therefore, each node is expected to receive the dataset, pre-process it and submit the score as well the deployment-ready model to the blockchain interface.

The different ML techniques employed by various nodes allow for modelling diverse use-cases such as :

1. Each UAV should have the flexibility to deploy an ML technique that is best suited for its purposes. Some UAVs would require an algorithm that would converge faster in a high probability cyber-attack environment. The UAV can also switch to a different algorithm as and when needed.

2. Different ML algorithms also allow for performing analytics and comparison within the ML techniques to gain insights on how well does the algorithm perform on certain data. The reason for deploying analytics.py node in the network is to generate the tables shown in Figure 9-14 .

4.6. Process flow:

The process starts with blockchain network initialization similar to Ethereum where a network ID is used however we use the network address of the monitor node as the network ID. We start the monitor node through a command line utility batch file similar to a Linux shell script. The monitor node looks for the *address.json* file where its address is maintained. This address is used by the rest of the blockchain network nodes to subscribe to the blockchain. In order to facilitate inter-node communication and off-chain sharing of machine learning models, we have used HTTP relying on the public internet for connectivity. Consequently, the participating nodes are implemented as web servers. Since our implementation is based on JSON-RPC micro services architecture, each miner has the endpoints that

can be requested for data and data can also be posted to them as well through GET and POST calls. To start the miner node, we run a command line utility batch file similar to the monitor node that searches for the address.json file that's present in the same directory as the miner node's code. This address.json file has two addresses; the address for the miner and the monitor node. Upon initialization, the miner node first sets own IP and port and then subscribes to the blockchain by performing the relevant JSON-RPC calls to the monitor node. After this step the miner runs its initialization procedures. It first submits its own address to the monitor node for it to be added in the network directory or global address space. Then it downloads the current up-to-date list of nodes that are part of the blockchain which ensures uniformity.

The verification of global address update can be done by fetching the address directory endpoint of any miner node. The same process is conducted for each node added to the network. The machine learning process is performed by first submitting the dataset to the blockchain network by uploading it to the monitor node which distributes the dataset to the participating nodes. After successful distribution of the dataset and its metadata to the nodes, the nodes commence the machine learning process. The conceptual model for the machine learning process is shown in Figure 4.

The machine learning process consists of the following 3 steps:

a. Data pre-processing: Each node submits the data and its associated metadata to the ML engine which performs the pre-processing on the data. In our implementation, this step comprises of reading the dataset and converting the dataset to machine-learning-ready format. The categorical columns are converted to numerical one-hot encoded columns and the numerical columns are normalized. Then training test split is applied to the data and model fitting is carried out. Specific steps to achieve pre-processing are highlighted below.

1. Columns are processed as per their types such as categorical or numerical. This is taken input on the blockchain node where the dataset is uploaded. The input form comprises of radio buttons on the target blockchain node's webpage that can be accessed by its IP. Once each column type is known, the next step is initiated which is one-hot encoding and normalization.

2. One-hot encoding is done for categorical columns which is the process of converting one column having more than two unique values into multiple columns having values of only 1 or 0. Normalization is done for numerical columns i.e. converting the values of different columns to a uniform range of values (0 to 1).
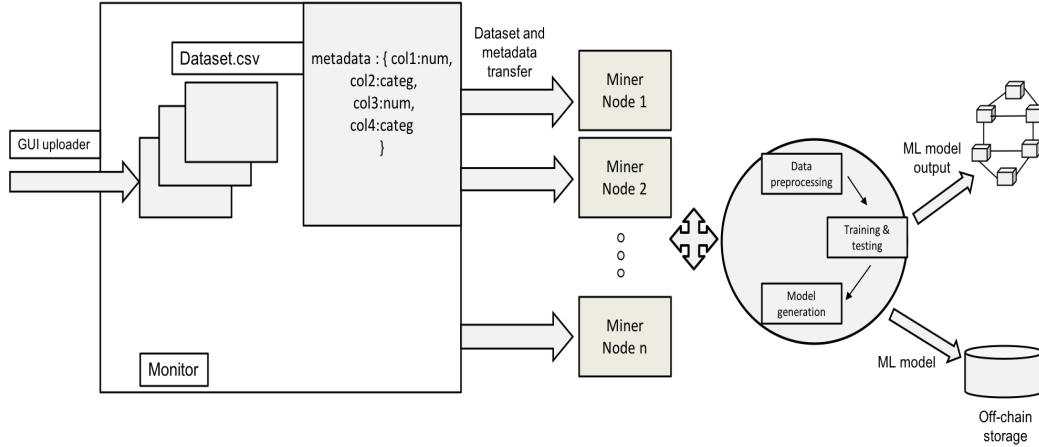
Figure 4: The conceptual model for the machine learning process

b. Model fitting and transaction formation: The model fitting is performed by using a suitable classifier algorithm. In our case, we have designated a set of classifiers that can be chosen at random by the ML engine and it is envisaged that different nodes running the machine learning engine will select a different algorithm thereby obtaining a different score. The different scores reported by each node will simulate a human-like crowd sourced machine learning process that will aid in prototyping a ranking/bench-marking policy enforced by the consensus algorithm governing the blockchain. When the machine learning classifier has completed its processing, the blockchain daemon converts this data to transaction and stores the transaction locally. It also broadcasts the same data to other miners to initiate the consensus process.

c. Consensus-based model sharing and block mining: After the individual nodes have completed the model fitting, each node has a copy of the ML model parameters of all the other nodes. In order to add the individual node data to blockchain a consensus algorithm is required such as Proof of Work adopted by Bitcoin. However, due to the nature of the application, we have adopted the ranking algorithm to achieve consensus for our setup. Therefore, a node is picked at random to add the block to the blockchain that contains reward for the winning nodes. This node first unifies the results from all nodes to ensure a fair selection of rewarded nodes and after a final consolidated ranking procedure is run, the block is mined by this node. The transaction contains the nodes and the rewards in case of multiple nodes achieving high scores.

4.7. System implementation:

In order to simulate successful transaction malleability attack, we have used *scikit-learn* Python libraries [73] following the major mathematical processes necessary for simulating basic characteristics of a typical blockchain network. These characteristics include

Figure 5: The webpage for the miner node

creation of genesis transaction, genesis block hash, honest and mutated transaction and their cryptographic schemes. Our implementation is comprised of a monitor node and three miner nodes all of which participate in the machine learning process. Table 1 presents the hardware specifications of the nodes used in our setup. The choice of the experimentation setup was motivated through availability of resources as well as their suitability for the experiments involved. For instance, as the nodes are required to store and share ML models on local storage, the storage capacity of the nodes was important which has been addressed by allocating 500GB for each node. Further, the RAM and CPU are comparable to similar experiments in existing literature however these machines were used solely for these experiments and no other software was running on them. In anticipation of the requirement to share machine learning models across participating nodes, the (off-chain) storage of blockchain and the network communication occurring between the nodes is achieved through python's flask web library.

As the miners are added to the blockchain network, they discover the rest of the nodes through the DNS mechanism outlined in 4.1. Each node runs a Apache Flask server and therefore has a webpage which can be used to view the transactions and the current state of the blockchain. A sample such output for a miner node is presented in Figure 5.

Node	CPU model	Clock speed	Memory	Storage	ML technique
Node 1	Intel Core i5	3.3 GHz	16 GB	500 GB	Onevsrest
Node 2	Intel Core i5	3.0 GHz	16 GB	500 GB	Logreg
Node 3	Intel Core i5	3.3 GHz	16 GB	500 GB	KNN
Node 4	Intel Core i5	3.3 GHz	16 GB	500 GB	Naive Bayes
Node 5	Intel Core i5	3.0 GHz	16 GB	500 GB	SGD

Table 1: Hardware specification for monitor and miner nodes

We initiated our blockchain with four nodes all of which participate in mining and machine learning process. In real-life scenario, each node is envisaged to receive data for analytics from an independent stream, however, for these experiments, we used the monitor/seed node to distribute data to other participating nodes. Upon receiving this data, each node performs initial anlaytics using its default machine learning algorithm. The outcome of this anlaytics (accuracy, TPR, FPR) is stored in the blockchain as a transaction whereas the resulting machine learning model is stored in the local storage of the node to facilitate off-chain storage and sharing. A sample transaction is presented in Figure6 whereas Figure 7 presents a sample block from our blockchain.

mine block request called block

{'index': 2, 'timestamp': '2020-04-15_22:01:18.211597', 'previous_hash': 'af1791
f0244e9b5d634ffe4eb74c5f82ecclebf8bf55c09a26dbea76a4d4d189', 'transactions': {'f
ilename': 'kd99_with_cols_shortened.csv', 'node_scores': ['score': 0.9983818770
226537, 'node': 127.0.0.1:5004', 'reward': 10), ('score': 0.9870550161812298,
node': '127.0.0.1:5010', 'reward': 9}, {'score': 9.9838187702265372, 'node': '12
7.0.0.1:5003', 'reward': 8}, {'score': 0.9627831715210357, 'node': '127.0.0.1:50
05', 'reward' 7], 'other_transaction_ids': [{'node': '127.0.0.1:5003', 'transa
ction_id':
'9670e66a92427b04a413eb62c813ee4e2bf844f9867646b2f34f7d2e4440133b'},
{'node': '127.0.0.1:5005', 'transaction_id': '075d4c9eab163ac5d65b4eac2a95793f16
d05b53dbab0996c68ea3e160657d66'}, {'node': '127.0.0.1:5010', 'transaction_id':'
94034f010c7aab9c9b1b9b954baaa7fe15fb7763b5aaf54c63c467358f89aecc'}]}}}
127.0.0.1 - - [15/Apr/2020 22:01:18] "<-[37mGET mine_block HTTP/1.1 <-[0m" 200 -
mine block request returned!

Figure 7: Sample block from the blockchain

transaction

{'datasetid': 'kd99_with_cols_shortened.csv', 'params': "KNeighbors Classifier(al
gorithm='auto', leaf_size=30, metric='minkowski',\n   metric_p
arams =None, n_jobs =None, n_neighbors =5, p=2,\n   weights='unif
orm')", 'node': '1dc32874e3bc4b46bf624ad6553ee2b6', 'node_ip': '127.0.0.1:5004',
  'pickle model': '127.0.0.1-5004-kd99_with cols shortened.csv.sav','score':9.9
983818770226537, 'transaction_id': '6f9e4ecdb6e74d9a7163750b1c9b499c9931d6918e7e
c249 a 4815fabaf664850', 'other transaction_ids': [{'node': '127.0.0.1:5003', 'tra
nsaction_id': '9670e66a92427b04a413eb62c813ee4e2bf844f9867646b2f34f7d2e4440133b'
> {'node': '127.0.0.1:5005', 'transaction_id': '075d4c9eab163ac5d65b4eac2a95793
f16d05b53dbab0996c68ea3e160657d66'}, {'node': '127.0.0.1:5010', 'transaction_id' : :
: '94034f010c7aab9c9b1b9b954baaa7fe15fb7763b5aaf54c63c467358f89aecc'}], 'sensiti
vity': 99.83818770226537, 'time': '0.46802687644958496', 'accuracy': 99.78308026
030369, 'precision': 99.83818770226537, 'fl': 99.83818770226537, 'ML_algo': 'knn
', 'stacking': None}
127.0.0.1 - - [15/Apr/2020 22:01:12] "<-[37mPOST /get_model HTTP/1.1 <-[0m" 200 -
mine block request called

Figure 6: Sample blockchain transaction

After a block is successfully added to the blockchain, each node node shares its machine learning with the other participants of the network so that this knowledge can be used to enhance their respective analytics. This is achieved by sharing models through off-chain storage and collaborative reinforcement is achieved by using these models via stacking technique. Similar to the first phase, the outcomes of this machine learning phase (accuracy, TPR, FPR) are added to the blockchain as a separate transaction.

5. Intrusion Detection Through Blockchain-based Decentralized ML

In order to evaluate the performance of the proposed blockchain-based decentralized machine learning system, we conducted experimentation using intrusion detection as a problem scenario. The motivation to choose intrusion detection is due to the significance of machine learning and artificial intelligence within this domain as highlighted by[74].

Our experimentation setup is based on four nodes all of which participate in machine learning and mining processes. Using KDD99 network-based attack dataset [75], we conduct analytics in two phases; first round is conducted independently at each node using data available at the node followed by the second round in which stacking is used to learn from the ML model created by other nodes of the blockchain network. A graphical representation of this process is presented in Figure 8.

5.1. KDD99 dataset

KDD99 is one of the most commonly used attack datasets to evaluate performance of intrusion detection systems [75]. The dataset consists of numerical and categorical features that describe the observations and states of a network connection. They are labelled from a set of 24 attack types which can be organized into four main categories of attacks. Furthermore, the dataset consists of 40 features which relate to basic attributes of TCP connection, traffic features and connection-oriented features. In table 2, we tabulate the details of some of the main features of the dataset belonging to each class.

- DOS: denial-of-service, e.g. sending a flood of SYN packets to the server so as to slow down the server's response to valid requests.
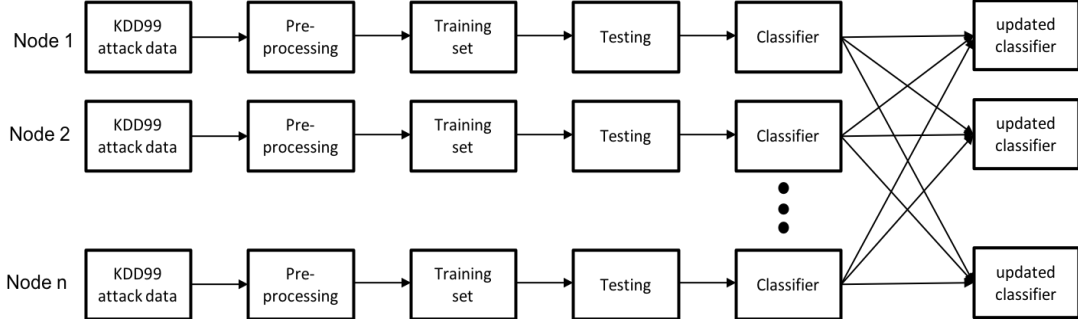
Figure 8: Overall framework for stacked ML

- R2L: unauthorized access from a remote machine, e.g. guessing password

- U2R: unauthorized access to local superuser (root) privileges, e.g., various "buffer overflow" attacks;

- probing: surveillance and other probing, e.g., port scanning.

5.2. Experimentation and Analysis

For our experimentation, a subset of the KDD99 dataset was used where each node was assigned 10,000 rows of data which was used for training and testing phases. The subset of data used contained 5 (most frequently occurring: *ipsweep, neptune, normal, portsweep, and smurf*) of the overall target classes in the main dataset. The class imbalance was chosen such that there is sufficient training data against each target class. The five target classes were:

As presented in Figure 8, the experiments were conducted in two phases. The first phase involved using machine learning independently at each node followed by refining the ML model by stacking models shared by other participating nodes of the blockchain network. The results of first phase (without stacking) are presented in table 3 whereas the outcome from the second phase of experiments (with stacking) are presented in table 4.

Overall, the experiments involved evaluating the system for the following metrics:

5.2.1. Accuracy

is defined as the percentage of correct predictions compared with total samples number of predictions is termed as accuracy in machine learning. Therefore, accuracy can be obtained by dividing the true positives by the total number of predictions that the classifier ran. Mathematically,

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

In our experiment, the classifier was tasked with predicting the correct attack type by labelling the test data with one of the 5 possible attack types used in our setup explained above. We then analyzed accuracy against each machine learning algorithm running on the nodes. Figure 9.A shows the scenario without stacking and Figure 9.B shows the scenario with stacking. Furthermore, Figure 9.C. presents a comparative analysis of performance of individual nodes (with respect to accuracy) with and without stacking to highlight the impact collaborative ML model sharing can achieve in such setting.

From the analysis of these graphics, we can observe a positive impact of ML stacking on the accuracy of classifiers i.e. the accuracy for most of the classifiers has increased by a range of 2 to 3 percent. The highest positive impact is observed for Naive Bayes algorithm when stacking is used whereas the smallest change is seen with KNN algorithm. An interesting observation here, is that the impact of stacked ML on accuracy can be limited because the base classifiers already reported a high prediction accuracy which, therefore, limits the potential for improvement. When we compare accuracy for other classifiers as shown in Figure 9.C, it is evident that nearly all classifiers saturate to a 99% accuracy after stacking is performed and all classifiers appear to be equal. The only factor that stands out differently when stacking is considered is the time to train with Naïve Bayes registering the least time to train in both non-stacking and stacking training runs.

5.2.2. Precision

Precision is a measure of finding out the extent of correctness of true positives. Precision is calculated by :

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

We plotted precision against each machine learning algorithm running on the nodes. Figure 11.A shows the scenario without stacking and Figure 11.B shows the scenario with stacking.

We also compared the two numbers for precision in both scenarios in a bar chart shown in Figure 11.C.

We can see that precision has experienced a 1 to 2 percent increase for most of the classifiers. The greatest improvement is again seen with Naive Bayes algorithm.

11

No.	Feature class	Feature name	Feature Description	Feature type
1	TCP connection attribute	duration	length (number of seconds) of the connection	continuous
2	TCP connection attribute	protocol_type	type of the protocol, e.g. tcp, udp, etc.	discrete
3	TCP connection attribute	service	network service on the destination, e.g., http, telnet, etc.	discrete
4	Connection domain specific attribute	num_failed_logins	number of failed login attempts	continuous
5	Connection domain specific attribute	num_root	number of "root" accesses	continuous
6	Connection domain specific attribute	is_guest_login	1 if the login is a "guest" login; 0 otherwise	discrete
7	Traffic specific attribute	count	no. of connections to the same host as the current connection in the past two seconds	continuous
8	Traffic specific attribute	serror_rate	% of connections that have "SYN" errors	continuous
9	Traffic specific attribute	same_srv_rate	% of connections to the same service	continuous

Table 2: Sample features of the KDD99 dataset

(a) Accuracy vs. Models

(b) Accuracy vs. Nodes

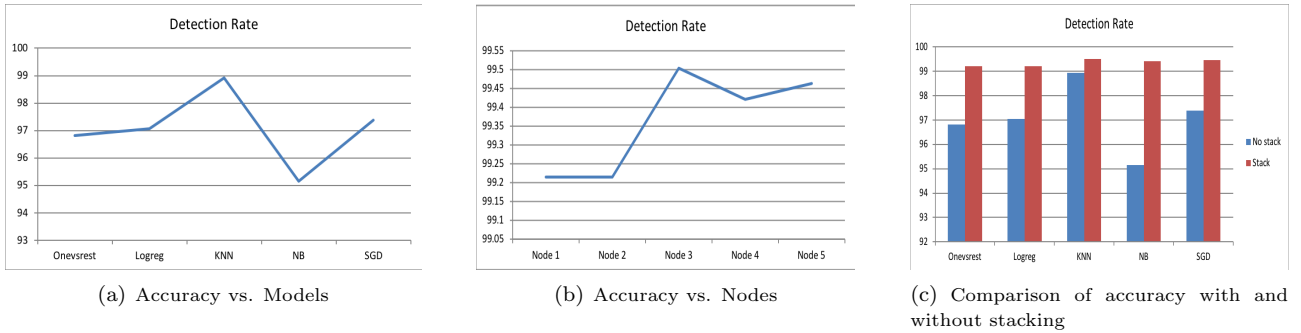(c) Comparison of accuracy with and without stacking

Figure 9: Accuracy of predictive analysis in different scenarios

The smallest change is seen with KNN algorithm same as accuracy. This is following the same pattern as that of accuracy. When we compare this metric with results obtained in the research work [76], the Naïve Bayes algorithm performs 85% whereas in our case, it shows a good 96%. This is further improved to 99% as we carry out stacking. As was seen in accuracy, the before and after stacking precision score for KNN hasn't changed much.

5.2.3. True Positive Rate

True positive rate (TPR) also known as detection accuracy represents the percentage of successful detection of malicious instances. TPR is calculated as:

$$TPR = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

Analyzing the outcomes of experiments with respect to TPR, interestingly, the results of the first phase (without stacking) are very similar to those of precision however a noticeable improvement in the TPR is identified in the phase (with stacking). The TPR for all the nodes witnessed improvement with node 4 achieving the highest detection accuracy of 99.6% demonstrating the effectiveness of the approach.

5.2.4. Time to Train

Time to train, is a metric we have decided to measure the performance in terms of speed of training. As the name implies, it is the time taken by the model to completely train on the data.

We plotted time to train against each machine learning algorithm running on the nodes. Figure 12.A shows the scenario without stacking and Figure 12.B shows the scenario with stacking. We also compared the two numbers for time to train in both scenarios in a bar chart shown in Figure 12.C.

Stacking requires machine learning done by several algorithms hence it is more time-consuming as is evident from Figure 12.C showing around 20 times increase for most of the classifiers. The shortest time to train is seen in Naïve Bayes algorithm. The longest time to train is seen with KNN algorithm. It is interesting to note that Naïve Bayes saturates to the same performance as KNN algorithm in case of stacking but with a quicker time to train.

5.2.5. Recall

is a measure that shows to what extent in the test set were the target classes accurately identified. If there are 100 positive classes in the test set and the classifier identifies 80 correctly and the rest 20 incorrectly than the
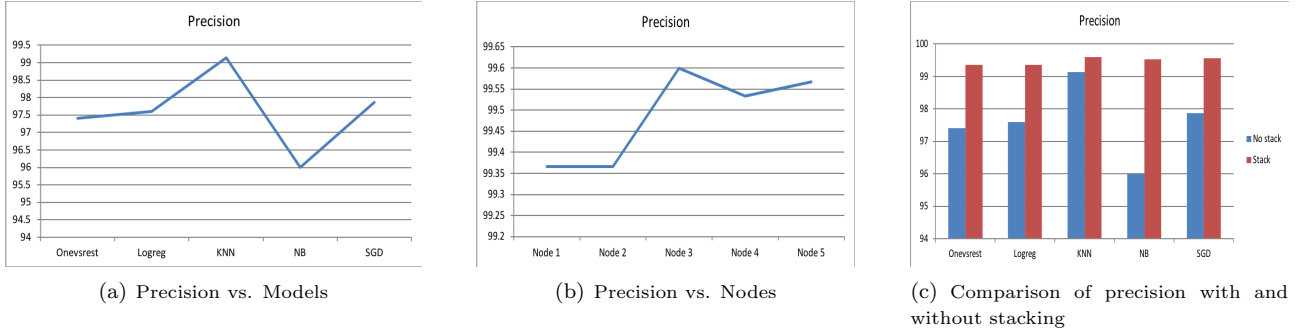
(a) Precision vs. Models

(b) Precision vs. Nodes

(c) Comparison of precision with and without stacking

Figure 10: Precision of predictive analysis in different scenarios

(a) TPR for different models

(b) TPR vs. nodes

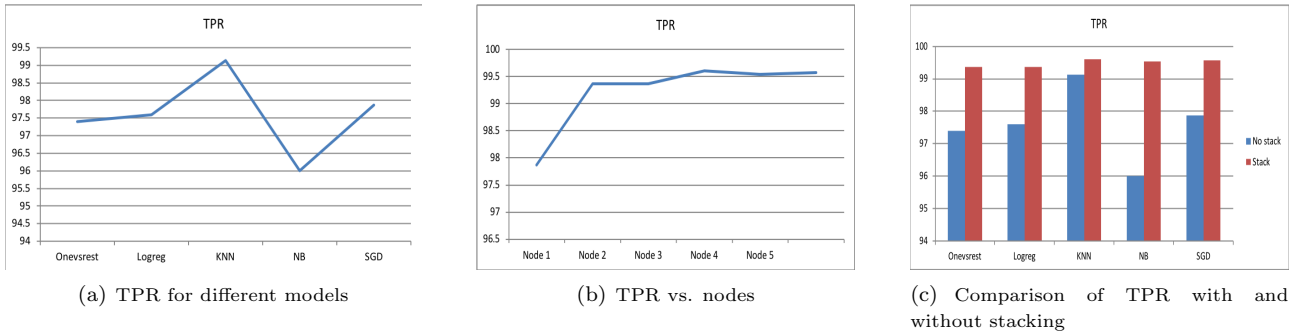(c) Comparison of TPR with and without stacking

Figure 11: TPR of predictive analysis in different scenarios

classifier accurately could recognize 80% of the true samples in the data correctly. Mathematically,

$$Recall = \frac{True\ Negative}{True\ Negative + False\ Positive}$$

We plotted recall against each machine learning algorithm running on the nodes. Figure 13.A shows the scenario without stacking and Figure 13.B shows the scenario with stacking.

We also compared the two numbers for recall in both scenarios in a bar chart shown in Figure 13.C.

Similar to precision, we can see that recall has experienced a 1 to 2 percent increase for most of the classifiers. Both Precision and Recall have been found to follow the same pattern for our case as is shown in Figure 13.C.

5.2.6. F1 score

is used to measure the balance between precision and recall for a machine learning classifier. It is calculated as,

$$F1\ score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

We plotted F1 score against each machine learning algorithm running on the nodes. Figure 14.A shows the scenario without stacking and Figure 14.B shows the scenario with stacking.

We also compared the two numbers for accuracy in both scenarios in a bar chart shown in Figure 14.C.

F1 score has also experienced an average 1 to 2 % increase for most of the classifiers. The greatest improvement is seen with Naive Bayes algorithm. The smallest change is seen with KNN algorithm. Much like the other metrics, F1 score has been found to be saturated around 99% when stacking is done.

It is seen that all algorithms saturate to 99% score whether in accuracy or precision and that is important to note because, in the case of stacking, one node is able to stack together all models from the other nodes as the models are actively exchanged among the nodes. However it can depend upon the criticality of the application where this is deployed because while giving an overall score boost, stacking decreases the time to train as well. These constraints should be taken into design time consideration taking their overall impact in question.

6. Conclusions and Future Outlook

UAVs or drones are being increasingly used in diverse application domains to facilitate improved operational efficiency and real-time decision making which require capability for intelligent processing of monitored data. Due to performance overheads, resource requirements of centralised machine learning techniques as well as the need for high quality predictive analytics in an efficient manner, decentralized machine learning methods are being explored. UAVs can especially benefit from these due to their inherent distributed architecture and
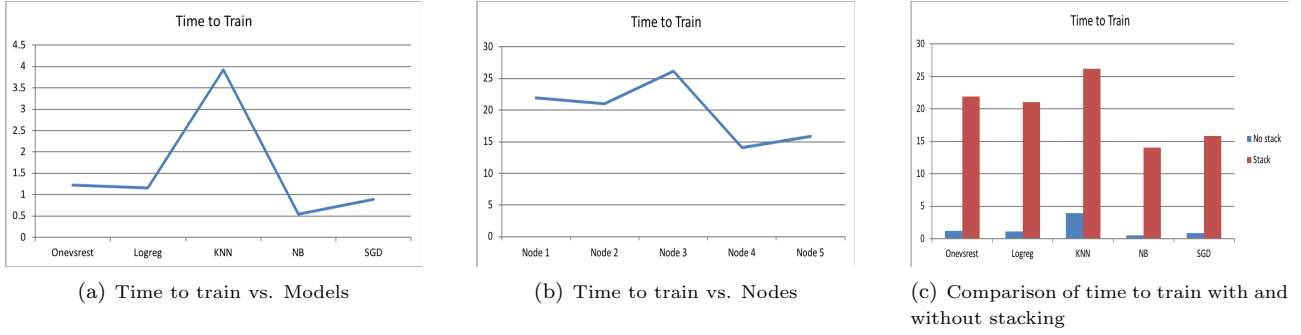
(a) Time to train vs. Models   (b) Time to train vs. Nodes   (c) Comparison of time to train with and without stacking

Figure 12: Time to train for predictive analysis in different scenarios

(a) Recall vs. Models   (b) Recall vs. Nodes   (c) Comparison of recall with and without stacking
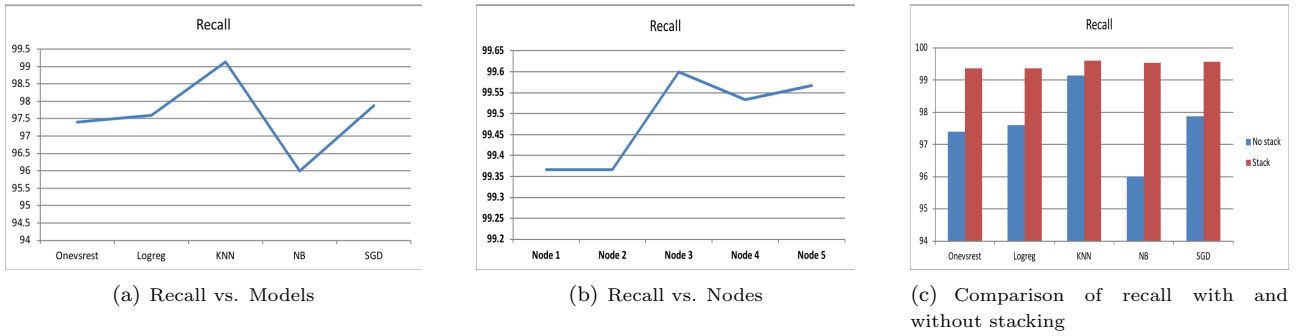
Figure 13: Recall for predictive analysis in different scenarios

No.	Algorithm	Accuracy	True Positive Rate	Precision	Recall	Sensitivity	Time to Train
1	Onevsrest	96.815	97.40	97.399	97.399	97.399	1.221
2	Logreg	97.057	97.60	97.600	97.600	97.600	1.154
3	KNN	98.928	99.133	99.133	99.133	99.133	3.928
4	NB	95.148	96	96	96	96	0.547
5	SGD	97.382	97.866	97.866	97.866	97.866	0.886

Table 3: Algorithms and measured machine learning metrics without stacking

Node	Base Algorithm	Accuracy	True Positive Rate	Precision	Recall	Sensitivity	Time to Train
1	Onevsrest	99.215	99.366	99.366	99.366	99.366	21.943
2	Logreg	99.215	99.366	99.366	99.366	99.366	21.049
3	KNN	99.503	99.60	99.600	99.600	99.600	26.191
4	NB	99.421	99.533	99.533	99.533	99.533	14.090
5	SGD	99.462	99.566	99.566	99.566	99.566	15.868

Table 4: Algorithms and measured machine learning metrics with stacking

constrained resource profile. Blockchain is a promising solution to shift traditional centralized approaches towards decentralisation with its immutable data structure, traceability and transparency of records, and fairness of use. Our proposed framework facilitate decentralized processing of machine learning based predictive analytics within a typical multi-UAV environment. Our proof of concept utilized stacking to achieve collaborative reinforcement of individual machine learning models to investigate its impact on efficiency of predictive analytics. We used intrusion detection as a case-study to evaluate the proposed system and have witnessed positive outcomes in terms of detection accuracy, TPR, FPR, F1 score and precision. The proposed framework has less performance overhead as compared to the centralised approach because instead of sharing data across participating nodes, machine learning models are shared across them so as to use this knowledge to improve accuracy of intelligent analytics across participating nodes. In addition, we achieve better privacy and security of data by removing the requirement of data sharing among the nodes.

In future we aim to explore the scalability of the proposed framework in the event of larger and disparate datasets, and greater number of participating nodes. Furthermore, it would be interesting to explore the robustness of the proposed framework in real world application specific scenarios of UAVs against variety of known attacks such as jamming or spoofing the navigational data, malicious code injection, Sybil attack
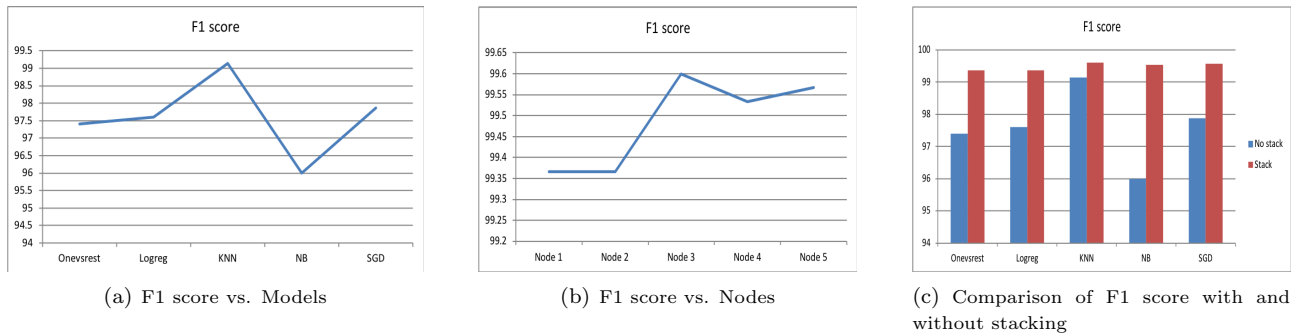
14

(a) F1 score vs. Models   (b) F1 score vs. Nodes   (c) Comparison of F1 score with and without stacking

Figure 14: F1 score of predictive analysis in different scenarios

and distributed denial of service attack.

References

[1] Abhishek Sharma, Pankhuri Vanjani, Nikhil Paliwal, Chathuranga M Wijerathna Basnayaka, Dushantha Nalin K Jayakody, Hwang-Cheng Wang, and P Muthuchidambaranathan. Communication and networking technologies for uavs: A survey. *Journal of Network and Computer Applications*, page 102739, 2020.

[2] Hassaan Hydher, Dushantha Nalin K Jayakody, Kasun T Hemachandra, and Tharaka Samarasinghe. Intelligent uav deployment for a disaster-resilient wireless network. *Sensors*, 20(21):6140, 2020.

[3] Fatma Outay, Hanan Abdullah Mengash, and Muhammad Adnan. Applications of unmanned aerial vehicle (uav) in road safety, traffic and highway infrastructure management: Recent advances and challenges. *Transportation research part A: policy and practice*, 141:116–129, 2020.

[4] Margaret Eichleay, Emily Evens, Kayla Stankevitz, and Caleb Parker. Using the unmanned aerial vehicle delivery decision tool to consider transporting medical supplies via drone. *Global Health: Science and Practice*, 7(4):500–506, 2019.

[5] Jeongeun Kim, Seungwon Kim, Chanyoung Ju, and Hyoung Il Son. Unmanned aerial vehicles in agriculture: A review of perspective of platform, control, and applications. *IEEE Access*, 7:105100–105115, 2019.

[6] Keunhyun Park and Reid Ewing. The usability of unmanned aerial vehicles (uavs) for measuring park-based physical activity. *Landscape and Urban Planning*, 167:157–164, 2017.

[7] Stefano Campana. Drones in archaeology. state-of-the-art and future perspectives. *Archaeological Prospection*, 24(4):275–296, 2017.

[8] Shiva Ram Reddy Singireddy and Tugrul U Daim. Technology roadmap: Drone delivery–amazon prime air. In *Infrastructure and Technology Management*, pages 387–412. Springer, 2018.

[9] Vinay Chamola, Vikas Hassija, Vatsal Gupta, and Mohsen Guizani. A comprehensive review of the covid-19 pandemic and the role of iot, drones, ai, blockchain, and 5g in managing its impact. *IEEE Access*, 8:90225–90265, 2020.

[10] Hazim Shakhatreh, Ahmad H Sawalmeh, Ala Al-Fuqaha, Zuochao Dou, Eyad Almaita, Issa Khalil, Noor Shamsiah Othman, Abdallah Khreishah, and Mohsen Guizani. Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges. *IEEE Access*, 7:48572–48634, 2019.

[11] Vikas Hassija, Vinay Chamola, Dara Nanda Gopala Krishna, and Mohsen Guizani. A distributed framework for energy trading between uavs and charging stations for critical applications. *IEEE Transactions on Vehicular Technology*, 69(5):5391–5402, 2020.

[12] Philip A Bernstein and Eric Newcomer. *Principles of transaction processing*. Morgan Kaufmann, 2009.

[13] Ioan Raicu, Ian Foster, Alex Szalay, and Gabriela Turcu. Astroportal: A science gateway for large-scale astronomy data analysis. In *Teragrid conference*, pages 12–15, 2006.

[14] Mouzhi Ge, Hind Bangui, and Barbora Buhnova. Big data for internet of things: a survey. *Future Generation Computer Systems*, 87:601–614, 2018.

[15] Li Da Xu and Lian Duan. Big data for cyber physical systems in industry 4.0: a survey. *Enterprise Information Systems*, 13(2):148–169, 2019.

[16] Kristof Coussement and Koen W. De Bock. Customer churn prediction in the online gambling industry: The beneficial effect of ensemble learning. *Journal of Business Research*, 66(9):1629 – 1636, 2013. Advancing Research Methods in Marketing.

[17] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. corr abs/1604.07316 (2016). *arXiv preprint arXiv:1604.07316*, 2016.

[18] Junaid Arshad, Muhammad Ajmal Azad, Mohammad Mahmoud Abdellatif, Muhammad Habib Ur Rehman, and Khaled Salah. Colide: a collaborative intrusion detection framework for internet of things. *IET Networks*, 8(1):3–14, 2018.

[19] Dario Amodei, Rishita Anubhai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Jingdong Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, et al. End to end speech recognition in english and mandarin. 2016.

[20] Mohamed Falah Faiz, Junaid Arshad, Mamoun Alazab, and Andrii Shalaginov. Predicting likelihood of legitimate data loss in email dlp. *Future Generation Computer Systems*, 2019.

[21] Farhan Riaz, Muhammad Ajmal Azad, Junaid Arshad, Muhammad Imran, Ali Hassan, and Saad Rehman. Pervasive blood pressure monitoring using photoplethysmogram (ppg) sensor. *Future Generation Computer Systems*, 98:120–130, 2019.

[22] Amir E Khandani, Adlar J Kim, and Andrew W Lo. Consumer credit-risk models via machine-learning algorithms. *Journal of Banking & Finance*, 34(11):2767–2787, 2010.

[23] Kevin Canini, Tushar Chandra, Eugene Ie, Jim McFadden, Ken Goldman, Mike Gunter, Jeremiah Harmsen, Kristen LeFevre, Dmitry Lepikhin, Tomas Lloret Llinares, et al. Sibyl: A system for large scale supervised machine learning. *Technical Talk*, 1:113, 2012.

[24] Ivan Kholod, Mikhail Kuprianov, and Ilya Petukhov. Distributed data mining based on actors for internet of things. In *2016 5th Mediterranean Conference on Embedded Computing (MECO)*, pages 480–484. IEEE, 2016.

[25] Elisa Bertino and Elena Ferrari. Big data security and privacy. In *A Comprehensive Guide Through the Italian Database Research Over the Last 25 Years*, pages 425–439. Springer, 2018.

[26] Diego Peteiro-Barral and Bertha Guijarro-Berdiñas. A survey of methods for distributed machine learning. *Progress in Artificial Intelligence*, 2(1):1–11, 2013.

[27] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321, 2015.

[28] H Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Aguera y Arcas. Federated learning of deep networks using model averaging. 2016.

[29] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 603–618, 2017.

[30] Rupa Ch, Gautam Srivastava, Thippa Reddy Gadekallu, Praveen Kumar Reddy Maddikunta, and Sweta Bhattacharya. Security and privacy of uav data using blockchain technology. *Journal of Information Security and Applications*, 55:102670, 2020.

[31] Tejasvi Alladi, Vinay Chamola, Nishad Sahu, and Mohsen Guizani. Applications of blockchain in unmanned aerial vehicles: A review. *Vehicular Communications*, 23:100249, 2020.

[32] Parimal Mehta, Rajesh Gupta, and Sudeep Tanwar. Blockchain envisioned uav networks: Challenges, solutions, and comparisons. *Computer Communications*, 151:518–538, 2020.

[33] Ilhaam Omar, Mazin Debe, Raja Jayaraman, Khaled Salah, Mohammed Omar, and Junaid Arshad. Blockchain-based supply chain traceability for covid-19 ppe. 2020.

[34] Dinesh Verma, Nirmit Desai, Alun Preece, and Ian Taylor. A block chain based architecture for asset management in coalition operations. In Tien Pham and Michael A. Kolodny, editors, *Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR VIII*, volume 10190, pages 223 – 231. International Society for Optics and Photonics, SPIE, 2017.

[35] Renming Qi, Chen Feng, Zheng Liu, and Nezih Mrad. Blockchain-powered internet of things, e-governance and e-democracy. In *E-Democracy for Smart Cities*, pages 509–520. Springer, 2017.

[36] Weizhi Meng, Elmar Wolfgang Tischhauser, Qingju Wang, Yu Wang, and Jinguang Han. When intrusion detection meets blockchain technology: a review. *Ieee Access*, 6:10179–10188, 2018.

[37] Muhammad Asaad Cheema, Muhammad Karam Shehzad, Hassaan Khaliq Qureshi, Syed Ali Hassan, and Haejoon Jung. A drone-aided blockchain-based smart vehicular network. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[38] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019. [https://dl.acm.org/doi/abs/10.1145/3298981].

[39] Benjamin CM Fung, Ke Wang, and S Yu Philip. Anonymizing classification data for privacy preservation. *IEEE transactions on knowledge and data engineering*, 19(5):711–725, 2007. [https://ieeexplore.ieee.org/abstract/document/4138206/].

[40] Steve Mansfield-Devine. Beyond bitcoin: using blockchain technology to provide assurance in the commercial world. *Computer Fraud & Security*, 2017(5):14–18, 2017.

[41] Yulei Wu, Hong-Ning Dai, Hao Wang, and Kim-Kwang Raymond Choo. Blockchain-based privacy preservation for 5g-enabled drone communications. *IEEE Network*, 35(1):50–56, 2021.

[42] Jian Wang, Yongxin Liu, Shuteng Niu, and Houbing Song. Lightweight blockchain assisted secure routing of swarm uas networking. *Computer Communications*, 165:131–140, 2021.

[43] Shubhani Aggarwal, Neeraj Kumar, Musaed Alhussein, and Ghulam Muhammad. Blockchain-based uav path planning for healthcare 4.0: Current challenges and the way ahead. *IEEE Network*, 35(1):20–29, 2021.

[44] F. Ahmad, Z. Ahmad, C. A. Kerrache, F. Kurugollu, A. Adnane, and E. Barka. Blockchain in internet-of-things: Architecture, applications and research directions. In *2019 International Conference on Computer and Information Sciences (ICCIS)*, pages 1–6, 2019.

[45] Kashif Mehboob Khan, Junaid Arshad, and Muhammad Mubashir Khan. Simulation of transaction malleability attack for blockchain-based e-voting. *Computers Electrical Engineering*, 83:106583, 2020.

[46] Kashif Mehboob Khan, Junaid Arshad, and Muhammad Mubashir Khan. Investigating performance constraints for blockchain based secure e-voting system. *Future Generation Computer Systems*, 105:13 – 26, 2020.

[47] Ahmed Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 IEEE symposium on security and privacy (SP)*, pages 839–858. IEEE, 2016.

[48] Zibin Zheng, Shaoan Xie, Hongning Dai, Xiangping Chen, and Huaimin Wang. An overview of blockchain technology: Architecture, consensus, and future trends. In *2017 IEEE international congress on big data (BigData congress)*, pages 557–564. IEEE, 2017.

[49] Suporn Pongnumkul, Chaiyaphum Siripanpornchana, and Suttipong Thajchayapong. Performance analysis of private blockchain platforms in varying workloads. In *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–6. IEEE, 2017.

[50] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konecny, Stefano Mazzocchi, H Brendan McMahan, et al. Towards federated learning at scale: System design. *arXiv preprint arXiv:1902.01046*, 2019. [https://arxiv.org/abs/1902.01046].

[51] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

[52] Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016. [https://arxiv.org/abs/1610.02527].

[53] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pages 315–323, 2013.

[54] Ohad Shamir, Nati Srebro, and Tong Zhang. Communication-efficient distributed optimization using an approximate newton-type method. In *International conference on machine learning*, pages 1000–1008, 2014.

[55] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. Scaling distributed machine learning with the parameter server. In *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*, pages 583–598, 2014.

[56] Tim Kraska, Ameet Talwalkar, John C Duchi, Rean Griffith, Michael J Franklin, and Michael I Jordan. Mlbase: A distributed machine-learning system. In *Cidr*, volume 1, pages 2–1, 2013.

[57] A Besir Kurtulmus and Kenny Daniel. Trustless machine learning contracts; evaluating and exchanging machine learning models on the ethereum blockchain. *arXiv preprint arXiv:1802.10185*, 2018. [https://arxiv.org/abs/1802.10185].

[58] C. Xu, K. Wang, and M. Guo. Intelligent resource management in blockchain-based cloud datacenters. *IEEE Cloud Computing*, 4(6):50–59, November 2017. [https://ieeexplore.ieee.org/abstract/document/8260822].

[59] Aissam Outchakoucht, ES Hamza, and Jean Philippe Leroy. Dynamic access control policy based on blockchain and machine learning for the internet of things. *Int. J. Adv. Comput. Sci.*

Appl., 8(7):417–424, 2017. [https://bit.ly/3eseN20].

[60] Dey Somdip. *A proof of work: Securing majority-attack in blockchain using machine learning and algorithmic game theory.* PhD thesis, Modern Education and Computer Science Press, 2018. [http://repository.essex.ac.uk/24024/].

[61] Xuhui Chen, Jinlong Ji, Changqing Luo, Weixian Liao, and Pan Li. When machine learning meets blockchain: A decentralized, privacy-preserving and secure design. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 1178–1187. IEEE, 2018.

[62] M. Ezuma, F. Erden, C. K. Anjinappa, O. Ozdemir, and I. Guvenc. Micro-uav detection and classification from rf fingerprints using machine learning techniques. In *2019 IEEE Aerospace Conference*, pages 1–13, 2019.

[63] I. Bisio, C. Garibotto, F. Lavagetto, A. Sciarrone, and S. Zappatore. Unauthorized amateur uav detection based on wifi statistical fingerprint analysis. *IEEE Communications Magazine*, 56(4):106–111, 2018.

[64] J. Kim, C. Park, J. Ahn, Y. Ko, J. Park, and J. C. Gallagher. Real-time uav sound detection and analysis system. In *2017 IEEE Sensors Applications Symposium (SAS)*, pages 1–5, 2017.

[65] A. Alipour-Fanid, M. Dabaghchian, N. Wang, P. Wang, L. Zhao, and K. Zeng. Machine learning-based delay-aware uav detection and operation mode identification over encrypted wi-fi traffic. *IEEE Transactions on Information Forensics and Security*, 15:2346–2360, 2020.

[66] Moayad Aloqaily, Ouns Bouachir, Azzedine Boukerche, and Ismaeel Al Ridhawi. Design guidelines for blockchain-assisted 5g-uav networks, 2020.

[67] Ezedin Barka, Chaker Abdelaziz Kerrache, Hadjer Benkraouda, Khaled Shuaib, Farhan Ahmad, and Fatih Kurugollu. Towards a trusted unmanned aerial system using blockchain (buas) for the protection of critical infrastructure. *Wiley Transactions on Emerging Telecommunications Technologies*, 2019.

[68] Tiago M Fernández-Caramés, Oscar Blanco-Novoa, Manuel Suárez-Albela, and Paula Fraga-Lamas. A uav and blockchain-based system for industry 4.0 inventory and traceability applications. In *Multidisciplinary Digital Publishing Institute Proceedings*, volume 4, page 26, 2018.

[69] Vishal Sharma, Ilsun You, Dushantha Nalin K Jayakody, Daniel Gutierrez Reina, and Kim-Kwang Raymond Choo. Neural-blockchain-based ultrareliable caching for edge-enabled uav networks. *IEEE Transactions on Industrial Informatics*, 15(10):5723–5736, 2019.

[70] . Retrieved from https://en.bitcoin.it/wiki/Bitcoin$_Core_0.11_(ch_3) :_I nitialization_a nd_S tartup.$[].

[71] BitCoin Community. Bitcoin Wiki. Retrieved from https://bitcoin.org/en/p2p-network-guidepeer-discovery. [].

[72] MultiChain. MultiChain Wiki. Retrieved from https://www.multichain.com/developers/creating-connecting/. [].

[73] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[74] Anna L Buczak and Erhan Guven. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications surveys & tutorials*, 18(2):1153–1176, 2015.

[75] Atilla Özgür and Hamit Erdem. A review of kdd99 dataset usage in intrusion detection and machine learning between 2010 and 2015. *PeerJ Preprints*, 4:e1954v1, 2016.

[76] G. Meena and R. R. Choudhary. A review paper on ids classification using kdd 99 and nsl kdd dataset in weka. In *2017 International Conference on Computer, Communications and Electronics (Comptelix)*, pages 553–558, 2017.